Scandinavian Journal of Information Systems

Volume 2 | Issue 1 Article 4

1990

APPROACHES TO APPRECIATE INFORMATION SYSTEMS METHODOLOGIES: A Soft Systems Survey

Peter A. Nielsen
University of Aalborg, Denmark, pan@cs.aau.dk

Scandinavian Journal of Information Systems, Vol. 2, pp. 43–60, 1990

APPROACHES TO APPRECIATE INFORMATION SYSTEMS METHODOLOGIES

A Soft Systems Survey

PETER A. NIELSEN
Department of Computer Science at University of Aalborg
Frederik Bajers Vej 7, DK-9220 Aalborg Ø, Denmark.

Abstract

A vast number of different methodologies for developing computer-based information systems are available. They form a motley spectrum in the sense that they apply to different parts of the development process and to different modes of thinking and acting. At a meta-level relative to methodologies is the discipline that deals with these differences and the applicability of methodologies in different situations. It encompasses theories, ideas and, especially, appreciation approaches, i.e. approaches to understand, evaluate and compare methodologies.

This article is a survey of the appreciation approaches found in the literature. The survey has been conducted as a soft systems enquiry. The appreciation approaches are understood and partly evaluated by mapping them unto a soft systems model that explicate how to answer the fundamental question of: Which methodologies in which situations?

It is argued that the appreciation approaches found are static and general in the sense that the fundamental question is assessed detached from a particular situation. Furthermore, it is argued on the basis of Schön's theory on reflection-in-action and Checkland's soft systems thinking that we need appreciation approaches that facilitate dynamic and situational thinking about the use of methodologies.

Keywords: information systems development, methodologies, appreciation of methodologies, soft systems.

1 Introduction

A methodology is not the same as practice (Mathiassen 1981, Andersen *et al.* 1990). *Working practice* is what is actually done in a situation in information systems development. It is the concrete actions taken. In contrast, a methodology is an abstraction, i.e. an intellectual construct prescribing actions to be taken.

An information systems *methodology* is taken to be a coherent set of guidelines. The guidelines prescribe actions both in terms of what to do and in terms of how to do it. The elements of 'how' are often refered to as techniques, that is precise concrete actions to be taken to achieve an end. Techniques are closely related to tools, models, and other kinds of means. Methodologies are basically abstract ideas about actions. Only when a methodology is interpreted and used can it influence the working practice of the actors seeking to apply it.

There is a wide variety of methodologies in information systems development. DeMarco (1979) has described a methodology (Structured Analysis/Structured Design) and it has later been elaborated on by Yourdon & Constantine (1979) and Yourdon (1982). It is probably the most well-known and widely used methodology. Using DeMarco's methodology information systems development is done by describing the existing and future information system in terms of data-processes, -items, and -flow. Jackson's (1983) JSD is a somewhat different methodology. Jackson advocates, instead, that entities in the information system is described as communicating sequential processes. ETHICS by Mumford (1983, 1987) is a methodology completely different from the two others. It does not prescribe how information systems should be described, but it provides guidance to how different interest groups should be involved and how to obtain job satisfaction for the users.

It is necessary for both practitioners and researchers to appreciate the differences between methodologies and to appreciate the applicability of methodologies. Firstly, a variety of methodologies already exist and it is not clear in the outset what distinguish them. Secondly, if a methodology is used in a situation where it is not appropriate it is likely to cause problems. If a professional choice between methodologies is going to be made some appreciation of the methodologies must be established.

During the last decade the study of information systems methodologies have taken various forms leading to many and quite different theories, ideas, and approaches to appreciate methodologies. Such studies are at a meta-level relative to methodologies. They have methodologies as the subject area whereas methodologies have information systems as their subject area. This article address the meta-level in the sense that it contains a survey and a discussion of approaches to appreciate methodologies. The *appreciation approaches* are ways of going about learning and understanding the features and the usefulness of methodologies used in information systems development. They are often, in the literature, denoted as approaches for 'evaluation of methodologies', 'comparison of methodologies', 'selection of methodologies', etc.

The survey is conducted as a soft systems enquiry. This means that the survey of the appreciation approaches is done by using Checkland's (1981) Soft Systems Methodology. Checkland's methodology is a general methodology for learning in unstructured situations; for a brief introduction see (Checkland 1985). It is based on soft systems ideas in the sense that systems are assumed *not* exist in the real world. Instead, we may enquire into the problems of the real world by formulating views on the real world in terms of systems, i.e. wholes, and then by comparing these systems views with the real world to learn about the real world.

Only one systems view is used in this survey. Many other views could have been used, but I have chosen a very simple and pragmatic view. Formulating it as a systems view forces me to be explicit about my choice. It is likely that others would have chosen differently. The point is, however, that the assertions made and the conclusions reached are all defensible against this view. By mapping the appreciation approaches that I have found in the literature onto the systems view it is possible to gain insight into these approaches. It is also possible to evaluate the approaches.

Section 2 contains the survey of the approaches to appreciate information systems methodology. Section 3 contains a general critique of the approaches found in the literature together with some ideas about a possible new approach. The concluding remarks are given in Section 4.

2 Approaches to Appreciate Methodologies

The fundamental question regarding appreciation approaches, "Which methodologies in which situations?", is chosen as the main focus in the survey. In terms of soft systems thinking this question can be formulated in a so-called root definition, see (Checkland 1981, p. 166–169, 221–227):

Root Definition Used in the Survey
A system for information systems researchers and practitioners to decide on which methodologies to seek to use in a specific situation based on knowledge about methodologies and information systems development.

The system describes a general abstract notion of a process of which the development and use of specific appreciation approaches are concrete examples.

Notice that the system does not exist in the real world, it is an explicitly formulated view-point. It is called a system because it is thought of as a whole with emergent properties, see (Checkland 1981, p. 74–82). The main emergent property of the above root definition is 'that a decision is reached', a property that does only apply to the system as a whole and not to any of its parts. It is assumed in the above root definition that such a decision can be reached by

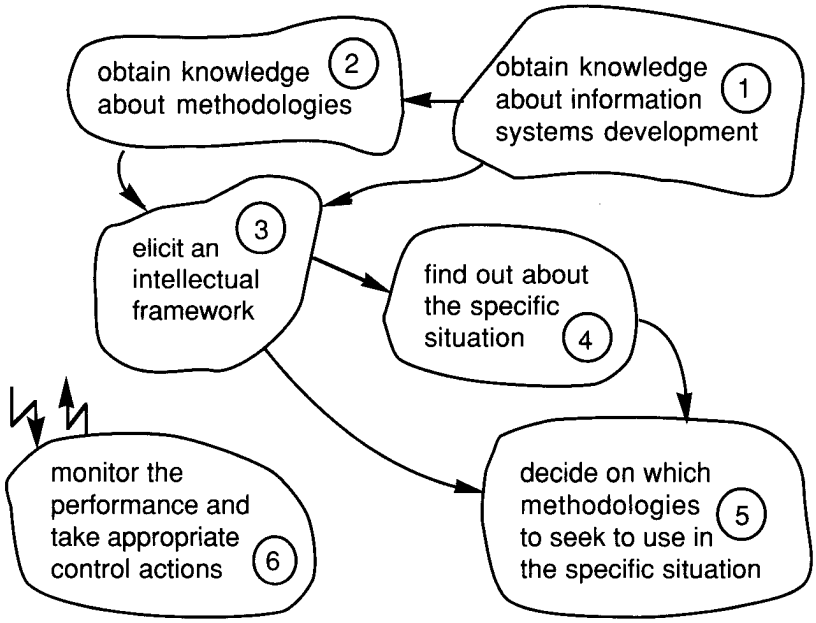

Figure 1: *The conceptual model used in the survey. It corresponds and is defensible against the above root definition. Key:* ◯*: activity;* →*: dependency between two activities;* ⇝*: dependency between an activity and all others.*

the involved actors if knowledge about methodologies and information systems development is available.

Corresponding to the root definition is the conceptual model, see (Checkland 1981, p. 169–177), used in the survey. This conceptual model is defensible against the root definition and it contains the minimal set of activities and their interdependencies necessary to achieve what is expressed in the root definition. The conceptual model corresponding to the choice of root definition is expressed in Figure 1 and it consists of six activities.

1. This activity is necessary because the root definition says: "based on knowledge about ... information systems development", thus the knowledge must be obtained.

2. This activity is necessary for the same reason as for (1). It depends on (1) because the methodologies we look at here are thought of as being useful in information systems development.

3. This activity is necessary in order to make the obtained knowledge into an intellectual form or framework that is relevant and useful in this particular system. Thus, it is highly dependent on (1) and (2).

4. This activity is necessary because the root definition says: "to use in a specific situation", therefore some finding out about what is special about a particular situation must be done. It depends on the outlook of the framework, that is, on the outcome of (3).

5. This activity is necessary for the obvious reason that, according to the root definition, a decision has to be taken. It depends on the outlook of the framework (3) and on what has been found out about the situation (4).

6. This activity is necessary to ensure that activities (1)–(5) are well performed by monitoring performance and taking appropriate control actions. Thus, in order to monitor all other activities it is dependent on these, and the control actions taken changes activities (1)–(5) making them all dependent on (6).

It is worth noticing that the activities and the dependencies only describe *what* necessarily must be done, but not how to do it.

Each of the appreciation approaches will be mapped onto the conceptual model. In order to do this, it is necessary to distinguish between two roles: the researcher and the practitioner. The researcher have developed an appreciation approach and the practitioner is using it. The mapping of each of the appreciation approaches unto the model is done to see *how* each of the activities has been performed by the researcher while developing the approach or *how* they are to be performed by the practitioner while using the approach.

For each of the appreciation approaches the detailed mapping is done by asking the following questions for each activity in the model:

α. If it has already been done by the researcher as part of developing the appreciation approach:

   1. How was it done?
   2. What was achieved?

β. If it is left for the practitioner to do as part of using the approach:

   1. Are there guidelines as to how it should be done?
   2. What is the desired outcome?

These questions are interesting because it is common to all the approaches in the survey that they relate to the fundamental question of 'Which methodologies in which situations?' one way or another. Of course, the researchers behind the approaches did not necessarily have this simple and pragmatic question in mind when they developed them; but the approaches can all be seen as though they seek to answer that question. In the following we will investigate to what extent and how well this question is answered in the literature. Since the conceptual model consists of the minimal set of activities necessary to answer the question

this investigation is done by assessing which activities have been covered by the researcher (α-questions) and which have not yet been covered and therefore left for the practitioner to do (β-questions).

2.1 The Mapping

The survey is primarily based on the three CRIS (Comparative Review of Information Systems Design Methodologies) proceedings: (Olle *et al.* 1982, 1983 and 1986) and on: (Tharp & Taggart 1977), (Ciborra *et al.* 1980), (Davis 1982), (Wood-Harper & Fitzgerald 1982), (Floyd 1983, 1984), (Episkopou & Wood-Harper 1986), (Jayaratna 1986), and (Benyon & Skidmore 1987), see also (Nielsen 1990, Chapter 1 and 3). The survey does not include appreciation approaches that look at methodologies *strictly* from the view-point of programming: (Bergland 1981) and (Davis 1988). I have chosen to let the survey deal with eight of these appreciation approaches.

Throughout this section the numbers, (x), refers to the activities in the conceptual model.

Taggart & Tharp

Taggart & Tharp (1977) have developed one of the earliest appreciation approaches found in the literature. They have what readily could be called a management information requirement analysis view on information systems development, (1). On the basis of this view and on a literature study of known methodologies in 1977, (2), they derive a framework, (3).

The framework consists of four categories: development process, information, decision making, and organisation. For each of the categories there is a number of key aspects, e.g. for organisation: organisation environment, organisation subsystems, and management function and level. Each of the considered methodologies is assessed for each of the key aspects on a scale from 1 to 3 expressing to what extent this aspect is treated. 1 means 'aspect not considered', 2 means 'recognition given to aspect', and 3 means 'significant treatment of aspect'. For example, in organisation environment: does the methodology recognise that the simplicity or complexity of information needs depend on the stability of the organisation's external environment and internal structure?

I find Taggart & Tharp's approach bold in the sense that each aspect must be assessed on a scale from 1 to 3. It is also a problem that they claim the framework to be comprehensive, cf. (Taggart & Tharp 1977, p. 275). It may be, of course, that this claim was believed to be true in 1977.

Brandt

The three CRIS conferences (Comparative Review of Information Systems Design Methodologies) is a sizable effort in comparing methodologies (Olle *et al.* 1982, 1983, 1986). The first conference was established around a test case concerned

with organising a conference (The IFIP Case) and each contributor was asked to apply a methodology to the test case. Each contribution was reviewed by a committee according to a large set of questions about methodologies in general and about how well they handled The IFIP Case. The purpose was that of taking stock and presenting a spectrum of methodologies. The second conference had the purpose of feature analysis of the methodologies presented at the first conference. A number of approaches for assessing the features came up. (Brandt 1983) was one of these.

In this approach, knowledge about methodologies has been obtained by studying how The IFIP Case had been handled, (2). Brandt arrives at a list of features that she finds important, (3). The seven features are: origin and experience, development process, data model, iteration and tests, representation means, documentation, user orientation, and tools and automation prospects. Having set up the list of features a comparison of methodologies is made.

The CRIS effort is remarkable, but also worrying. More than anything it shows the difficulty in developing one common test case and then hope that the methodologies under investigation are suitable for that case. For example, it is assumed in The IFIP Case that a computer system is going to be developed, its requirements are fixed, there are no users to interact with, etc. What if you have a methodology particularly suited for defining the problems of the organisation in a participative process? In Brandt's appreciation approach I find that the comparison is very superficial. This may have been caused by the CRIS setting, but it may just as well have been caused by the lack of referenced insight, theoretical or practical, into information systems development.

Wasserman, Freeman & Porcella

Wasserman *et al.* (1983) provide an overview of a great number of methodologies in order to see to what extent they can be integrated with the programming language Ada. Based on a theoretical exploration and reasoning from the viewpoint of information systems development as software development, (1), they have designed a questionnaire about methodologies. The knowledge about methodologies they have obtained stems from the questionaires answered by those who developed each of the methodologies, (2). Twenty-four of the responses refer to methodologies actually in use and thus hoping to draw upon experience with the use of the methodologies.

The intellectual framework behind their overview consists of six areas of concern, (3): life cycle coverage, applicability, technical concepts supported, workproducts and representation, quality assurance, and usage. The overview of the twenty-four methodologies provides insight into how each of them deals with the six areas.

An obvious objection against this appreciation approach is that it relies fully and uncritically on experience gained by those who developed the methodologies. Wasserman *et al.* are aware of this, but it is almost like trusting a car sales-man.

A strength of this approach is that it seeks to gain the overview from an explicit and precise viewpoint, i.e. program development.

Floyd

Floyd (1984, 1986) has studied the application of a few methodologies in practice. The methodologies have been studied, (2), by "teaching courses in which the methods were presented, tried out by students on a case study and subsequently evaluated" (Floyd 1986, p. 20). Based on a software engineering view on information systems development, (1), and the experience gained through the laboratory experiments a framework has been elicited, (3).

Floyd offers a few concepts for categorisation of methodologies as an intellectual framework. The concepts are: area of application, perspective, guidelines, theory, coherence, coverage, and product-oriented features. The categorising concepts are not used very much by Floyd as she prefers the concrete experience with methodologies. A typical claim is: "the notion of 'action' [in JSD] is confusing since it has no time dimension" (Floyd 1986, p. 30).

It is a problem with Floyd's approach, I find, that it lacks theoretical coherence in the framework. There is seemingly nothing to tie the characterising concepts together and thereby provide an argument as to why these concepts are useful. Floyd argues, to some extent, that her framework is intentionally vague in the direction of categorising and setting a taxonomy for methods as her purpose is more to clarify the area of methodologies as no suitable criteria for a taxonomy can be found. At this point, Floyd is using the obvious lack of an objective criteria as an argument for not being precise and consistent from a particular theoretical viewpoint. On the other hand, Floyd's investigation provides much substantial and useful knowledge about the actual use of methodologies.

Ciborra, Bracchi & Maggiolini

Ciborra *et al.* (1980) are most likely the first to seek to relate the differences of methodologies with differences of situations in information systems development. As they put it: "different task environments imply different problem spaces and, above all, different methods and approaches" (Ciborra *et al.* 1980, p. 52).

Their appreciation approach is based on Newell & Simon's view on problem solving and theoretical work on information systems development, (1). The knowledge about methodologies has been gained through a literature study, (2). It is from this theoretical standpoint and with methodologies in mind that they arrive at their intellectual framework, (3).

In the framework, a situation, or task environment as they call it, is seen as consisting of four characteristics: organisation and information system, technology, users and systems developers, and, finally, project management. To each characteristic there is a measure. For example, for users and systems developers it is twofold: 'Are the analysts dependent on involvement with the users or can they

work detached?' And: 'Do the users act passively or actively?' Methodologies can be assessed according to what characteristics they can cope with.

As part of the approach it is claimed that it is possible to assess the characteristics of a specific situation, (4). From this assessment, an appropriate methodology is found, (5). The choice is simple since it is only necessary to find a methodology which can cope with the characteristics of the situation.

A weakness of this approach is, as I see it, that it is not at all clear how each of the characteristics should be measured, e.g., whether the users are active or passive may not always be measurable with sufficient significance. On the other hand, the characteristics provide some insight into the relationship between situations and methodologies in a simple and logical way.

Davis

Davis's (1981) approach for matching methodologies and situations is probably the most well-known in its field. It is based on Simon's work on human information processing and Davis's application of this on information systems development, (1). A number of factors that influence the level of uncertainty are identified.

Knowledge about methodologies, (2), stems primarily from a literature study (though an exception is a comparison of two methodologies based on practice (Munro & Davis 1977)). The framework is established, (3), as a match between the uncertainty factors and how much uncertainty a methodology can cope with. The match is a single scale of uncertainty. This means that each methodology can be categorised according to how much uncertainty it can cope with. At the lowest level of uncertainty the category is named 'asking', then in rising order: 'deriving from existing system', 'traditional analysis', and 'experimentation'. The methodologies mentioned in the introduction of this article all belong to traditional analysis.

The level of uncertainty of a specific situation is determined by a three step procedure, (4). First, the situation is characterised by organisational context, information system, users, and systems developers. Second, the uncertainty of the development process is evaluated in terms of existence of requirements, users's ability to express requirements, and systems developers's ability to understand requirements. Third, the overall uncertainty is evaluated. Thereafter, a primary and secondary category of methodologies is selected and within this one or more methodologies is selected, (5).

One of the weaknesses of Davis's approach is that it is difficult to evaluate the uncertainty factors and hence to make convincing evaluations. Another weakness is that it is not at all clear that uncertainty is sufficient as the one and only measure. On the other hand, the strength of the approach is that a systems developer in a particular situation will be forced to think about what undoubtedly are, relevant aspects of the situation and by that find arguments for a decision on which methodologies to seek to use.

Episkopou & Wood-Harper

Wood-Harper & Fitzgerald (1982) developed a taxonomy of methodologies. They claim that methodologies fall into six categories: general systems theory, human activity systems, participation, traditional, data analysis, and structured systems methodologies. (It is striking that the category general systems theory is without content.)

Later, Episkopou & Wood-Harper (1986) have taken the taxonomy and Checkland's work on problem content solving systems and developed an appreciation approach they call a framework for choosing appropriate methodologies. The approach is based on action research involving more than 60 systems developers (Episkopou & Wood-Harper 1986, p. 227). Even if there is no explicit evidence it seems to be from these research efforts that the insight into information systems development and methodologies is gained, (1), (2).

The framework itself, (3), is structured according to Checkland's problem contents solving system where the basic distinction made is between problem owner, problem solver, problem contents system, and problem solving system. The approach establishes which factors it is important to consider when matching situations and methodologies. For the problem owner and problem solver, it is factors like cognitive style, skills, and ability to specify. For the problem content system, it is factors like resources and interest groups. As part of the framework, methodologies are classified according to the underlying ideology, tolls provided, inquiring system, manpower and time needed.

In order to learn about a specific situation, (4), assessment of the problem owner, the problem solver, and the problem content system is performed. After this, the problem solving system is formulated by choosing one or more methodologies, (5), that are appropriate to the assessment.

One of the weaknesses of Episkopou & Wood-Harper's approach is that they have not established a reasonable relationship between the situational factors and the features of methodologies. They are, of course, aware of this as well as the fact that the appreciation approach has not been tried in practice. A strength of the approach is the conceptual clarity of the areas to consider carefully, i.e. problem owner, problem solver, etc.

Jayaratna

Jayaratna's (1986) approach towards methodologies and situations is called Normative Information Model-based Systems Analysis and Design (NIMSAD). Basically, it is a framework for understanding and evaluating methodologies and their use in information systems development.

NIMSAD has been developed and used over a period of years and by this experience has been gained with methodologies, information systems development, and NIMSAD itself, (1), (2), (3). The framework itself is a model of information systems development with eight stages.

1. Introduction to the 'real world'.
2. Understanding the situation of concern.
3. Diagnosis.
4. Prognosis outline.
5. Systems analysis.
6. Logical design.
7. Physical design.
8. Implementation.

Together with these eight stages a few methodologies have been evaluated by finding out how they support each of the stages.

When it comes to understanding a specific situation and deciding which methodology to use, (4), (5), NIMSAD is unusual. It is unusual in the sense that it is not a framework which provides guidelines for performing (4) and (5) once and for all in a project, e.g. when the project is established. On the contrary, it is a framework that is intended to guide the systems developer all the way through a project by providing opportunity 'at a conscious level of concern' to (re-)evaluate and (re-)select methodologies, (6).

The strength of Jayaratna's approach is twofold, at least. First, it has shown its usefulness through practice. That cannot be said by any other approach in the survey. Second, it is worth noticing that the approach has something to say about activity (6), i.e. the monitor and control activity. The approach encourages the systems developer to re-evaluate and re-selecte which involves re-doing acitivities (2), (3), (4), (5). A weakness of the approach is that it is difficult to distinguish phases like diagnosis, prognosis outline and systems analysis. In practice, they are seldom carried out in that order, and they are likely to be inseparable in action and in logic.

2.2 Summary of the Mapping

The most significant differences between the appreciation approaches can be summarised by clustering the mapping onto activities (1), (2), (3) and (4), (5), see Figure 2. This leaves out activity (6), but only (Jayaratna 1986) deals with that anyway.

The summary in Figure 2 reveals four categories of appreciation approaches, see Figure 3. The theory-based approaches are based on theoretical research efforts in activities (1), (2) and (3) while the practice-based are based on practical research efforts. The approaches that deal with methodologies alone cannot be mapped onto activities (4) and (5) while the approaches that deal with both

	(1), (2), (3): obtain knowledge about ISD and methodologies and elicit framework	(4), (5): find out about specific situation and decide on methodologies	Strengths and weaknesses
Taggart & Tharp	A management view on ISD and literature study of methodologies → a survey of methodologies	n/a	(−): bold measures
Brandt	A list of features of methodologies and the IFIP Case → a comparison of methodologies	n/a	(−): superficial comparison and the artificiality of The IFIP Case
Wasserman *et al.*	A programming view on ISD and questionaires about methodologies → an overview of methodologies	n/a	(+): explicit and precise viewpoint behind overview (−): rely on authors's own claims
Floyd	Laboratory experiments with methodologies → experience with methodologies	n/a	(+): substantial insight (−): too little coherence in framework
Ciborra *et al.*	A Simonean view on ISD and literature study of methodologies → a match of task environment characteristics with methodologies	Determine characteristics of present situation	(+): a simple relationship between situations and methodologies (−): unclear measures
Davis	A Simonean view on ISD and literature study of methodologies → a match of uncertainties of situations with methodologies	Determine uncertainty of present situation	(+): forced to think through relevant aspects (−): difficult to believe in uncertainty measure
Episkopou & Wood-Harper	Action research → factors of importance	Assess problem owner, problem solver, problem content system and formulate problem solving system	(+): conceptual clarity (−): no relationship between situations and methodologies
Jayaratna	Used in practice → a model of ISD relevant to understand methodological practice	(Re-)evaluate and (re-)select methodologies throughout a project	(+): it is useful and the framework can be redone from within (−): difficult to distinguish phases

Figure 2: *The mapping of the appreciation approaches in summary. Key: →: leads to; n/a: not applicable.*

	Theory-based	Practice-based
Methodologies	(Taggart & Tharp 1977) (Brandt 1983)	(Wasserman *et al.* 1983) (Floyd 1984 & 1986)
Methodologies and situations	(Ciborra *et al.* 1980) (Davis 1982)	(Episkopou & Wood-Harper 1986) (Jayaratna 1986)

Figure 3: *Four categories of appreciation approaches.*

methodologies and situations have something to say about how to do (4) and (5).

It is a characteristic of all the approaches found in the literature that they are stable and general, the only exception being (Jayaratna 1986). They are stable in the sense that the frameworks are unadaptive. They are general in the sense that unique situational knowledge is disregarded. That is not necessarily a quality. What if a new methodology is developed and its features cannot be captured by means of a stable appreciation approach? What if information systems development changes likewise? What if systems developers in a particular situation possess substantial knowledge about methodologies that they want to apply systematically and this is impossible by means of a general appreciation approach. In the following, I shall argue that there are indeed good reasons for developing appreciation approaches which are dynamic rather than stable and situational rather than general.

3 From General Approaches to Situational Approaches

Schön (1983) has made a thorough analysis of how practitioners think and act. The analysis is about the thinking and acting of professional practitioners in general. It can, however, easily be argued that systems developers can be understood as professional practitioners, too.

Schön distinguishes between two different modes of thinking: technical rationality and reflection-in-action. Technical rationality is seen as instrumental problem solving. The practitioner takes a goal as given and by selecting the best means seeks to reach this goal (Schön 1983, p. 21ff). Hence, technical rationality is based on the assumption that there is consensus about the goal and that the goal is visible and clear. The selection of best means is done by applying the relevant scientific theory (Schön 1983, p. 34). Knowledge is mainly a result of science and can be separated from practice, where professional knowledge is

	Technical rationality	Reflection-in-action
Situations	Falls into scientifically defined categories	Are unique, complex, uncertain, and value-conflictual
Knowledge	Is a result of science and must be separated from practice	Is inseparable from action
Practice	Is fundamentally different from research (science); practice is application of theory and research is production of theory	Includes research

Figure 4: *Significant differences between the two view-points on professional practice.*

seen as consisting of three types: an underlying basic science, an applied science, and skills and attitudes. In this sense practice and science (research) remain separate. Knowledge produced by research is specialised and standardised theories and techniques that can be applied by practitioners to diagnose situations and solve problems. Practical knowledge is knowledge about the relationship of means to ends.

In contrast to technical rationality, reflection-in-action is seen as problem setting where each situation is considered unique: "the practitioner approaches the practice problem as a unique case ... seeks to discover the particular features of his problematic situation, and from their gradual discovery, design an intervention" (Schön 1983, p. 134). The three most significant differences between technical rationality and reflection-in-action are shown i Figure 4.

Schön discusses why technical rationality must be abandoned and why we must see professional practice as reflection-in-action. Firstly, a situation is fundamentally unique:

> Even when a problem has been constructed, it may escape the categories of applied science because it presents itself as unique or unstable. In order to solve a problem by the application of existing theory or technique, a practitioner must be able to map those categories onto features of the practice situation. ... But a unique case falls outside the categories of applied theory; an unstable situation slips out from under them. (Schön 1983, p. 41)

Secondly, taking a goal as given ignores problem setting. Schön argues that in practice problems are not given, they are constructed from the problematic situation. The process of setting the problem is:

> a process in which, interactively, we *name* the things to which we will attend and *frame* the context in which we will attend to them. (Schön 1983, p. 40)

That is, that ends are not given, ends and means are mutually dependent and is therefore defined in the same process.

Let us now return to the appreciation approaches with Schön's theory in mind. The approaches in the survey all take the stance of technical rationality to some extent. All the approaches define a stable framework where the concepts that distinguish methodologies are found during research and never changed later on. In this sense they rely on the assumption that knowledge is a result of research and that systems developers involved in practice merely apply the knowledge already established. That is, they assume a strong divison between research and practice. The approaches that deal with both situations and methodologies all take the stance that situations and methodologies can be categorised. This means, in terms of Schön, that situations fall into well-defined categories and having determined which category a specific situation belongs to the optimal methodology can be chosen. In other words, the approaches are general in the sense that the uniqueness of a situation is disregarded. Jayaratna's approach is a slight exception from this because it allows and almost encourages the practitioner to modify parts of the framework and hence move in direction of reflection-in-action.

Schön would very much like to abandon the whole idea of technical rationality. I shall, however, merely claim that the appreciation approaches belonging to technical rationality are insufficient. They are useful but insufficient. They are useful because it is possible and useful to investigate the features of methodologies and situations. If, for example, experience from practice shows that DeMarco's data-flow technique is not useful when describing issue-based work in a bank, because it is not in logic nor in practice routines, then this experience is generalisable into knowledge about data-flow detached from the situation where it was experienced.

On the other hand, the approaches are insufficient because the general argument put forward by Schön does apply to the use of methodologies in information systems development. From this point of view we must strive at practicing reflection-in-action in the process of matching methodologies and situations. The consequence is that appreciation approaches must *also* take into account that situations are unique and substantial knowledge is gained through practice by systems developers. This calls for appreciation approaches which are more dynamic and situational. The work ahead lies in finding out how to do the *dynamic and situational thinking* without loosing the general methodological knowledge of existing approaches.

The idea I will bring about here is to base such an approach on Soft Systems Methodology (Checkland 1981). Soft Systems Methodology is a systems approach for problem solving or, more rightly, for learning in unstructured situations. It is close to the process of reflection-in-action in the sense that when using it situations are seen as unique and insight into the uniqueness is gained.

The starting point of Soft Systems Methodology is an unstructured problematic situation. The perceived problems are not taken as given, but rather as a rich expression of the nature of the situation is sought. Having expressed the richness, several root definitions are formulated as systems views that are potentially relevant to learn about the situation. The root definitions denote views of the situation and are not accounts of the real world as it is. For each root definition a conceptual model is built. The conceptual models are then compared with the situation. Based on the comparison of several conceptual models with the situation, feasible and desirable changes are found. The conceptual models structure a debate amongst the actors. By this the actors define for themselves which changes to bring into action.

Now, if the problematic situation has to do with finding out which methodology to seek to use we can use Soft Systems Methodology. Relevant views on the situation are modelled as systems views. Within this, it is possible to take the existing general methodological knowledge as relevant views of the situation. The methodological knowledge possessed by actors in the situation is taken to be views as well. Formulating root definitions and building conceptual knowledge of these general and local (situational) views enables a comparison and debate about which methodologies to seek to use.

4 Conclusion

I started out in this article stating explicitly the standpoint from which I wanted to survey the literature on appreciation approaches. The standpoint was formulated as a root definition and detailed as a conceptual model. The survey was then made by mapping each of the approaches onto the conceptual model. This highlighted to what extent and in what way each of the approaches has answered the question: Which methodologies in which situations? It also illustrated some of the limitations and qualities of each of the approaches.

By means of Schön's work on reflection-in-action I argued that the surveyed approaches, except Jayaratna's, have taken the stance of technical rationality in the sense that they are stable and general. I also argued that this stance is insufficient and that there is a need for appreciation approaches that take the stance of reflection-in-action in the sense of dynamic and situational. This should be done without abandoning technical rationality, but by utilising the best of it.

By means of Checkland's work I suggested that one place to search for such an approach is within soft systems thinking. Within this theoretical and practical tradition we may simultaneously utilise existing knowledge about methodologies

and situations and establish new methodological insight through a process of dynamic and situational thinking.

References

Andersen, N. E., *et al.*, (1990). *Professional Systems Development.* Prentice-Hall, New York.

Benyon, D. & S. Skidmore, (1987). Towards a tool kit for the systems analyst. *The Computer Journal*, 30(1):2–7.

Bergland, G. D., (1981). A guided tour of program design methodologies. *Computer*, (October):13–37.

Brandt, I., (1983). A comparative study of information systems design methodologies. In (Olle *et al.* 1983).

Checkland, P. B., (1981). *Systems Thinking, Systems Practice.* John Wiley, Chichester.

Checkland, P. B., (1985). Achieving 'desirable and feasible' change: An application of Soft Systems Methodology. *Journal of the Operational Research Society*, 36(9): 821–831.

Ciborra, C., G. Bracchi & P. Maggiolini (1980). A multiple-contingency review of systems analysis methods and models. In H. C. Lucas, F. Land, Lincoln, and Supper, editors, *The Information Systems Enviroment*, Amsterdam, North-Holland, pages 47–60.

Davis, G. B., (1982). Strategies for information requirements determination. *IBM Systems Journal*, 21(1):4–30.

Davis, A. M., (1988). A comparison of techniques for the specification of external systems behavior. *Comm. ACM*, 31(9):1098–1115.

DeMarco, T., (1979). *Structural Analysis and Systems Specification.* Yourdon Inc. & Prentice-Hall, Englewood Cliffs, N. J.

Episkopou, D. & A. T. Wood-Harper, (1986). Towards a framework to choose appropriate IS approach. *The Computer Journal*, 29(3):222–228.

Floyd, C., (1984). A comparative evaluation of system development methods. Seminar paper, Technische Universität Berlin, Institut für Angewandte Informatik, Berlin, March. Seminar at Dansk Datamatik Center, Denmark.

Floyd, C., (1986). A comparative evaluation of system development methods. In (Olle *et al.* 1986).

Iivari, J. & P. Kerola, (1983). A sociocybernetic framework for the feature analysis of information systems design methodologies. In (Olle *et al.* 1983), pages 87–139.

Jackson, M., (1983). *Systems Development.* Prentice Hall, London.

Jayaratna, N., (1986). Normative information model-based systems analysis and design (NIMSAD): A framework for understanding and evaluating methodologies. *Journal of Applied Systems Analysis*, 13:73–88.

Mathiassen, L., (1981). *Systemudvikling og systemudviklingsmetode, (Systems Development and Systems Development Method.* Technical Report PB-136, Computer Science Department, Århus University.

Mumford, E., (1983). *Designing Human Systems for New Technology. The ETHICS METHOD.* Manchester Business School, Manchester.

Mumford, E., (1987). Socio-technical Systems Design — Evolving Theory and Practice. In Bjerknes, G., P. Ehn & M. Kyng, editors. *Computers and Democracy.* Gower, Aldershot.

Munro, M. C. & G. B. Davis, (1977). Determining management information needs - a

comparison of methods. *MIS Quarterly*, 1(2).

Nielsen, P. A., (1990). *Learning and Using Methodologies in Information Systems Analysis and Design.* Ph.D. Thesis, Dept. of Systems and Information Management, University of Lancaster.

Olle, T. W., H. G. Sol & A. A. Verrijn-Stuart, editors, (1982). *Information Systems Design Methodologies: A Comparative Review*, Amsterdam, IFIP, WG 8.1, North-Holland.

Olle, T. W., H. G. Sol & C. J. Tully, editors, (1983). *Information Systems Design Methodologies: A Feature Analysis*, Amsterdam, IFIP, WG 8.1, North-Holland.

Olle, T. W., H. G. Sol & A. A. Verrijn-Stuart, editors, (1986). *Information Systems Design Methodologies: Improving the Practice*, Amsterdam, IFIP, WG 8.1, North-Holland.

Schön, D., (1983). *The Reflective Practitioner - How Professionals Think in Action.* Basic Books, New York.

Taggart, W. & M. O. Tharp, (1977). A survey of information requirements analysis techniques. *Computing Survey*, 9(4):273–290.

Wasserman, A. I., P. Freeman & M. Porcella, (1983). Characteristics of software development methodologies. In (Olle *et al.* 1983).

Wood-Harper, A. T. & G. Fitzgerald, (1982). A taxonomy of current approaches to systems analysis. *The Computer Journal*, 25(1):12–16.

Yourdon, E., (1982). *Managing the Systems Life Cycle.* Yourdon Inc., New York.

Yourdon, E. & L. L. Constantine, (1979). *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design.* Prentice-Hall, Englewood Cliffs, N. J.